Модифицированный алгоритм поиска светлячками

(реализация предложенной модификации и разработка своей)

Было предложено реализовать алгоритм поиска светлячками из следующей статьи. Этот алгоритм должен решать оптимизационную задачу – поиска оптимальных значений параметров функции, чтобы её значение было максимальным или минимальным. То есть, производится поиск глобального максимума или минимума.

Оригинальный алгоритм поиска светлячком был представлен в 2009 году и имитирует поведение светлячков ночью. Этот алгоритм берет за основу следующие правила:

1. Светлячки не имеют пола, то есть каждый из них может быть привлечён к любому другому
2. Яркость светлячка определяется значением оптимизируемой функции
3. Привлекательность светлячка для других пропорциональна его яркости, но уменьшается с увеличением расстояния до него (в соответствии с коэффициентом поглощения света)
4. Каждый светлячок двигается к наиболее яркому, которого видит. Если таких светлячков нет, он двигается случайно.

То есть, движение каждого светлячка определяется следующим образом:

(считается, что каждого светлячка определяет набор координат – вектор – N-мерном пространстве функции – значения её соответствующих параметров).

$$x \coloneqq x + A_0 e^{-\beta r^2}(x' - x) + \alpha\varepsilon$$

, где:

$A_0 -$ привлекательность светлячка

$\alpha -$ параметр случайности, определяет длину шага

$\varepsilon -$ случайный единичный вектор

$\beta -$ коэффициент поглощения света среды

В оригинальной версии алгоритма A_0 для простоты вычислений принято считать единицей.

То есть, алгоритм в общем случае описывается так:

1. Сгенерировать случайный набор светлячков (векторов)
2. Вычислить яркость для каждого из них

3. Двигать каждого светлячка к другим более ярким (если таких нет, двигать его случайно)
4. Обновить яркость для каждого светлячка новым значением
5. Прекратить выполнение если достигнут критерий останова, иначе вернуться к шагу 2

В статье предлагается модифицировать алгоритм следующим образом:

1. В случае, когда при движении светлячка более ярких не находится, двигать его не случайно, так как это может уменьшить значение функции, привязанное к нему. Предлагается же генерировать набор из m случайных единичных векторов и двигать такого светлячка только в направлении, которое увеличит значение функции, привязанное к нему. Если же такого единичного вектора (направления) не найдётся, не двигать светлячка вообще.
2. Предлагается не считать значение A_0 равным единице. Авторы статьи считают, что лучше будет вычислять это значение в зависимости от яркости двух соответствующих светлячков следующим образом:

$$A_0 = \frac{I_0{}'}{I_0}$$

, где:

I_0 – яркость светлячка, к которому двигается текущий светлячок

$I_0{}'$ – яркость светлячка, который двигается к более яркому

(*обращается внимание на то, что в случае $I_0 = 0$ необходимо вычислять A_0 как-то иначе, например как $e^{I_0{}' - I_0}$, главное, чтобы значение A_0 было пропорционально яркости источника $I_0{}'$*)

Тестирование алгоритма проводилось на семи функциях. Рассмотрим каждую из них:

1. **Вторая функция Де Джонга**

Требуется найти максимум следующей функции при следующих ограничениях:

$$\max \quad f_1 = -\left(20 + \left(x_1^2 - 10\cos 2\pi x_1\right) + \left(x_2^2 - 10\cos 2\pi x_2\right)\right),$$

$$\text{s.t.} \quad -2.048 \le x_1, \quad x_2 \le 2.048.$$

Это известный тест для алгоритмов оптимизации – функция имеет очень много локальных оптимумов и уникальный глобальный оптимум 0 в точке (0;0).

2. Функция Есома

Требуется найти максимум следующей функции при следующих ограничениях:

$$\begin{aligned}&\max \quad f_2 = (\cos x_1 \cos x_2)\, e^{-(x_1-\pi)^2-(x_2-\pi)^2},\\ &\text{s.t.} \quad -20 \le x_1,\ x_2 \le 20.\end{aligned}$$

Это оптимизационная задача с единственным глобальным решением – большая площадь плоской поверхности этой функции и единственный оптимум равный 1 в точке $(\pi;\pi)$ делают её сложной.

3. Прерывистый тест

Требуется найти максимум следующей функции при следующих ограничениях:

$$\begin{aligned}&\max \quad f_3 = \sum_{i=1}^{2} \operatorname{int}(x_i),\\ &\text{s.t.} \quad -5.12 \le x_1, \quad x_2 \le 5.12.\end{aligned}$$

Оптимальное значение равно 10 и достигается при $x_i \ge 4.5$ для $i = 1, 2$. Эта оптимизационная задача сложна тем, что если не задать подходящую длину шага для алгоритма (α), многие алгоритмы застревают в одной из плоских поверхностей функции, а не двигаются к лучшему решению.

4. Функция Мишры

Требуется найти максимум следующей функции при следующих ограничениях:

$$\begin{aligned}&\max \quad f_4 = \ln\left\{\left[\left(\sin(\cos x_1 + \cos x_2)^2\right)^2 - \left(\cos(\sin x_1 + \sin x_2)^2\right)^2 + x_1\right]^2\right\}\\ &\qquad\qquad - 0.1\left((x_1-1)^2 + (x_2-1)^2\right),\\ &\text{s.t.} \quad -10 \le x_1, \quad x_2 \le 10.\end{aligned}$$

Это мультимодальная функция со значением оптимума равным 2.28395 в точке (2.8863;1.8233).

5. Стохастический тест

Требуется найти максимум следующей функции при следующих ограничениях:

$$\max \quad f_5 = 5e^{-(x_1-\pi)^2-(x_2-\pi)^2} + \sum_{j=1}^{2}\sum_{i=1}^{2}\varepsilon_{ij}e^{-(x_1-i)^2-(x_2-j)^2},$$

$$\text{s.t.} \quad 0 \le x_1, \quad x_2 \le 10,$$

Она сложна тем, что включает в себя случайный параметр шума ε_{ij} – случайное число от 0 до 1. Оптимум этой функции находится в точке $(\pi;\pi)$ и находится в промежутке от 5 до 5.0794 в зависимости от параметров шума, использованных при вычислении. Из-за этого параметра шума мы вряд ли получим одни и те же результаты при разных запусках алгоритма.

6. Функция Стеблинского

Требуется найти максимум следующей функции при следующих ограничениях:

$$\max \quad f_6 = 280 - \left(\frac{x_1^4 - 16x_1^2 + 5x_1}{2} + \frac{x_2^4 - 16x_2^2 + 5x_2}{2}\right),$$

$$\text{s.t.} \quad -5 \le x_1, \quad x_2 \le 5.$$

Это мультимодальная функция с оптимумом равным 358.3323 в точке (-2.9035; -2.9035).

7. Функция Шуберта

Требуется найти максимум следующей функции при следующих ограничениях:

$$\max \quad f_7 = -\left[\sum_{i=1}^{5} i\cos\left((i+1)\,x_1 + i\right)\right]\left[\sum_{i=1}^{5} i\cos\left((i+1)\,x_2 + i\right)\right],$$

$$\text{s.t.} \quad -10 \le x_1, \quad x_2 \le 10.$$

У этой оптимизационной задачи есть 760 оптимумов, 18 из которых – глобальные со значениями равными 186.73.067.

Результаты тестирования представленной в статье модификации на приведённых выше функциях показаны в первых двух секциях блокнота. Модифицированная версия алгоритма реализована на Python с использованием Numpy и Numba и на C с упаковкой в dll, которая позже запускается из Python-кода с использованием модуля ctypes.

Я предлагаю собственную модификацию этого алгоритма – модификацию Фурлеба. Предлагается сделать следующие изменения:

Отказаться от принципа бесполости светлячков – поделить их на мальчиков и девочек по следующему принципу:

a. Светлячки-мальчики стремятся к максимуму функции
b. Светлячки-девочки стремятся к минимуму функции
c. Между светлячками одного пола взаимодействие происходит тем же образом, что и в ранее рассмотренной модификации – девочки движутся к другим менее ярким девочкам, а мальчики – к другим более ярким мальчикам. Если менее или более (соответственно) яркого сородича не находится, соответствующий светлячок пытается найти направление движения, которое уменьшит или увеличит его яркость соответственно. Если же и такого направления не находится, светлячок остаётся на месте.
d. Между светлячками разного пола взаимодействие происходит иначе: мальчики стремятся избегать девочек, двигаясь в противоположном направлении – тем дальше, чем выше отношение их яркостей. Для светлячков-девочек аналогично

Таким образом увеличивается скорость сходимости алгоритма к лучшему решению и, как побочный продукт его работы, находится минимальное значение рассматриваемой функции и значения параметров, при которых оно достигается.

Для того, чтобы регулировать соотношение мальчиков и девочек в популяции светлячков нужно ввести новый параметр в алгоритм – коэффициент разлада:

$$discordance = \frac{C_{boys}}{C}$$

, где:

C_{boys} – число мальчиков в популяции светлячков

C – общее число светлячков в популяции

Реализация предложенной модификации и её тестирование (обычное и сравнительное) показаны в третьей секции блокнота.

При сравнительном тестировании добавляются новые тестовые функции:

1. **Функция-сфера:**

Требуется найти максимум следующей функции (умноженной на -1) при следующих ограничениях:

$f_6 x = \sum_{i=1}^{n} x_i^2$	256	[−5.12, 5.12]

У этой оптимизационной задачи единственный оптимум – 0. Сложность этого теста заключается в большом количестве измерений функции.

2. **Функция-косинус:**

Требуется найти максимум следующей функции (умноженной на -1) при следующих ограничениях:

$f_9 x = \sum_{i=1}^{n} x_i^2 - 10\cos 2\pi x_i + 10$	30	[−5.12, 5.12]

У этой оптимизационной задачи единственный оптимум – 0.

Исходя из результатов тестирования, хотя обе модификации не всегда доходят близко к глобальному максимуму, модификация Фурлеба показывает себя значительно лучше.